
Demonstration Applications

ACIS is shipped with two demonstration applications that illustrate ACIS functionality. These applications can be used for becoming familiar with ACIS, testing ideas, or creating prototypes for an application. These demonstration applications can also be used for reporting problems to *Spatial's* technical support department.

Scheme ACIS Interface Driver Extension (Scheme AIDE) is a Scheme based ACIS demonstration application. It accepts Scheme commands that are entered in a command window and displays the results in a separate view window. Refer to Chapter 7, *Using Scheme AIDE*, for more information on using Scheme AIDE.

The ACIS Test Harness is an unsupported demonstration application, written in C++, with a simple command-line interface. It accepts English-like commands that are entered in a command window and displays the results in a separate view window. Refer to Chapter 8, *Using the Test Harness*, for more information on using the Test Harness application.

Developers can create their own commands that can be used with these programs to test ideas during their own application development.

What You Need to Know

This section describes the things you need to know in order to work effectively with ACIS. It covers three areas:

- | | |
|----------------------------------|---|
| <i>Prerequisites</i> | Summarizes the foundation you should have <i>before</i> you start working with ACIS. |
| <i>ACIS Fundamentals</i> | Summarizes the fundamental ACIS concepts with which you should first become familiar. |
| <i>Scheme Fundamentals</i> | Summarizes the fundamental Scheme concepts with which you should first become familiar if you are going to work with ACIS using Scheme. |

Prerequisites

Before you begin working with ACIS, you should have the following foundation:

- An understanding of geometric modeling
- An understanding of object oriented design and programming
- Familiarity with an object oriented programming language, such as C++

You may develop your ACIS based applications with either C++ or Scheme. Therefore, prior programming experience with one of these languages is extremely helpful.

ACIS Fundamentals

You should become familiar with the ACIS concepts and terminology described in the documentation and with the ACIS API functions and C++ classes before working with the modeler. ACIS addresses a variety of modeling issues across a wide range of potential industries and applications. Therefore, many developers do not need to know *everything* about the modeler in order to complete their particular development projects. However, an understanding of a minimum set of subjects is required to begin working with the modeler.

The following list summarizes the minimum set of ACIS fundamental subjects you should understand before you start to develop your application:

- ACIS software architecture, including the object libraries
- Geometry
 - mathematical C++ classes (such as position, vector, transf)
 - construction geometry C++ classes (such as curve, sphere, surface, etc.)
 - model geometry C++ classes (such as CURVE, SPHERE SURFACE, etc.)
- Model topology
 - boundary representation
 - topology C++ classes (such as BODY, EDGE, FACE, etc.)
- Entities and the ENTITY class
- Model objects and class relationships
- Attributes
 - types of attributes
 - C++ class ATTRIB and classes derived from it
- History and roll
 - bulletin board
 - delta state
 - history stream

- Save and restore
 - save file format
 - C++ class ENTITY_LIST

These subjects are discussed in these manuals (as well as in various reference templates):

- *ACIS Getting Started Guide*
- *ACIS Application Development Manual*
- Component manuals, including the *Kernel Component Manual*

The C++ functions and classes are described in reference templates in online help.

Scheme Fundamentals

If you plan to develop your application in whole or in part with the Scheme language, or to use the Scheme AIDE demonstration application, you should also understand the following Scheme subjects, particularly as they relate to ACIS:

- Scheme language
- Scheme data types
- Scheme procedures
- Scheme extensions

Scheme is discussed in the *Scheme Support Component Manual*. The Scheme data types and Scheme extensions are described in reference templates in online help.

3